FSD Laboratory 04

Name - Shounak Dighe
Roll No - 20
Pannel - H
Prn - 1032233107

Aim: Write server-side script in PHP to perform form validation and create database application using PHP and MySQL to perform insert, update, delete and search operations.

Objectives:

1. To understand Server-side Scripting.
2. To learn database connectivity using PHP-MySQL.
3. To perform insert, update, delete and search operations on database.

Theory:

PHP Architecture

1. Client-Side Request:

- A user requests a PHP page through a web browser.

2. Web Server:

- The web server (like Apache or Nginx) receives the request. It knows that the request is for a PHP file and forwards it to the PHP engine.

3. PHP Engine:

- The PHP engine processes the PHP script. This engine could be built into the web server (like with PHP-FPM) or run as a module (like with mod_php).

4. Code Execution:

- The PHP engine executes the PHP code, which may include database queries, business logic, and other operations.

5. Output Generation:

- The PHP script generates HTML (or other content) as output, which is then sent back to the web server.

6. Web Server Response:

- The web server sends the generated content back to the user's web browser for display.

Steps for Database Connectivity in PHP

1. **Choose a Database:**

- Decide which database system you want to use (e.g., MySQL, PostgreSQL).

2. **Set Up the Database:**

- Install and configure the database server.
- Create a database and user with appropriate permissions.

3. **Connect to the Database:**

- Use PHP functions or classes to establish a connection to the database. Here's a basic example using MySQLi and PDO:

Using MySQLi:

```
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "database";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
```

Using PDO:

```
$dsn = "mysql:host=localhost;dbname=database";
$username = "username";
$password = "password";

try {
    $conn = new PDO($dsn, $username, $password);
    // Set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
} catch(PDOException $e) {
    echo "Connection failed: " . $e->getMessage();
}
```

4. **Execute Queries:**

- Write and execute SQL queries using the connection.

5. **Close the Connection:**

- Close the connection when done to free up resources.

Using MySQLi:

```
$conn->close();
```

Using PDO:

```
$conn = null;
```

FAQ:

1. What are the advantages of Server-side Scripting?

- **Security:** Server-side scripts are executed on the server, so sensitive code and database interactions are not exposed to the client.
- **Database Interaction:** It allows direct interaction with databases, enabling dynamic content generation based on user input or other factors.
- **Consistency:** The server processes scripts in a consistent environment, reducing variations in how scripts execute on different client devices.
- **Control:** Server-side scripting provides better control over the content served to users, including the ability to manage user sessions and data processing.

2. What is XAMPP and phpMyAdmin?

- **XAMPP:** XAMPP is a free and open-source cross-platform web server solution stack package. It includes Apache (web server), MySQL (database), PHP (server-side scripting), and Perl. It simplifies the process of setting up a local server environment for development purposes.
- **phpMyAdmin:** phpMyAdmin is a free and open-source tool written in PHP intended to handle the administration of MySQL over the web. It provides a web interface to interact with MySQL databases, allowing users to manage databases, tables, columns, and execute SQL queries through a user-friendly GUI.

3. What are the two ways to connect to a database in PHP?

- **MySQLi (MySQL Improved):** A PHP extension that provides an interface to communicate with MySQL databases. It supports both procedural and object-oriented programming.

- **PDO (PHP Data Objects):** A database access layer providing a uniform method of access to multiple databases. PDO supports multiple database systems and provides an object-oriented interface.

Problem Statements:

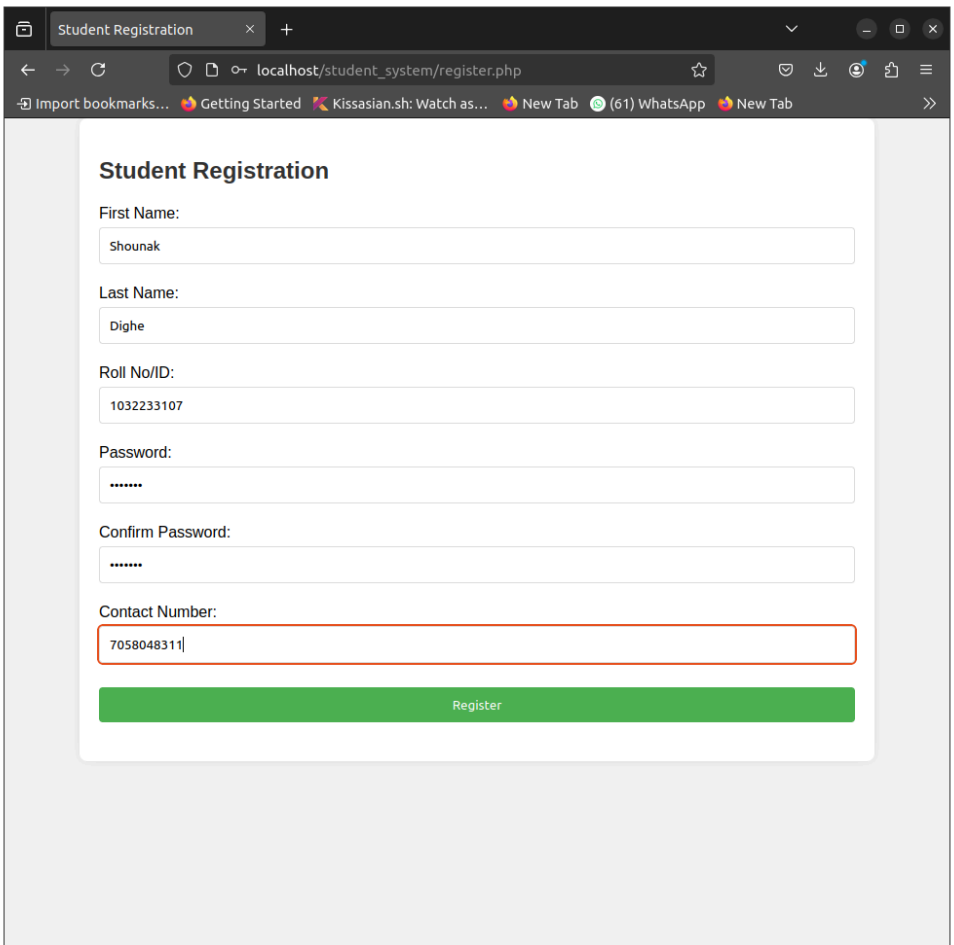
PHP CRUD Operations

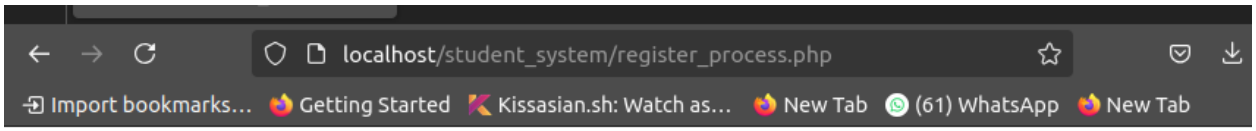
1.Student can create a PHP form or use existing/ implemented HTML form for Student's Registration System with the fields mentioned: First name,Last name, Roll No/ID, Password, Confirm Password,Contact number and perform following operations

1.Insert student details -First name,Last name, Roll No/ID, Password, Confirm Password,Contact number
2.Delete the Student records based on Roll no/ID
3.Update the Student details based on Roll no/ID- Example students can update their contact details based on searching the record with Roll no.
4.Display the Updated student details or View the Students record in tabular format.

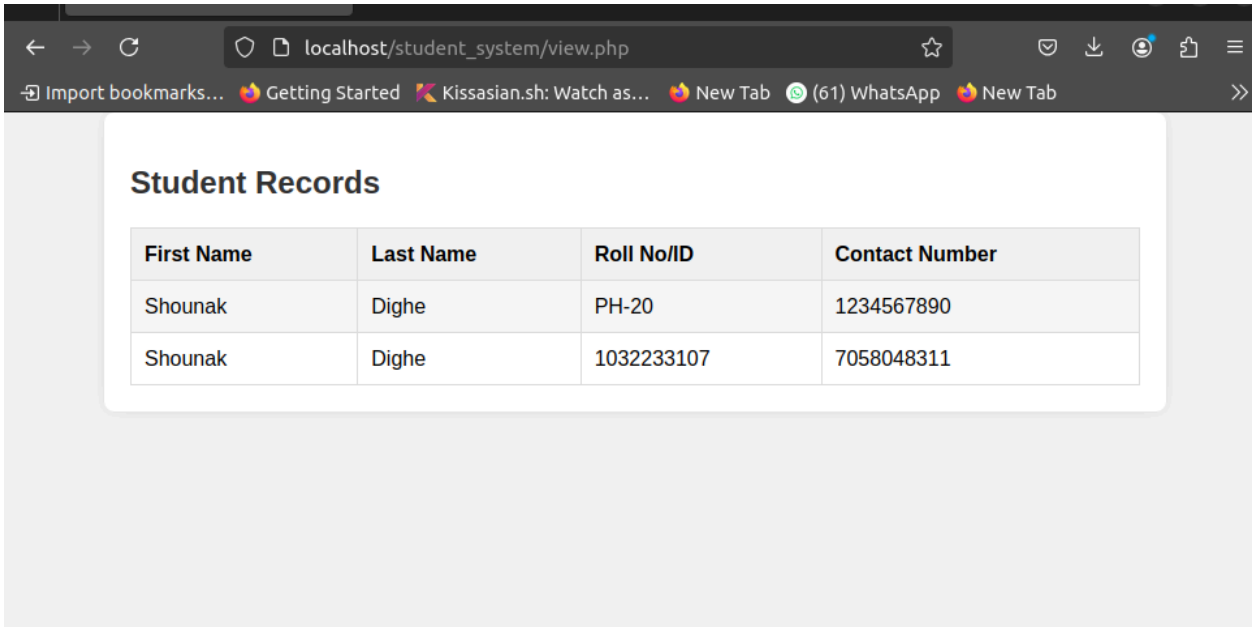
Apply Form Validation on the necessary fields using PHP/Javascript

Student Registration

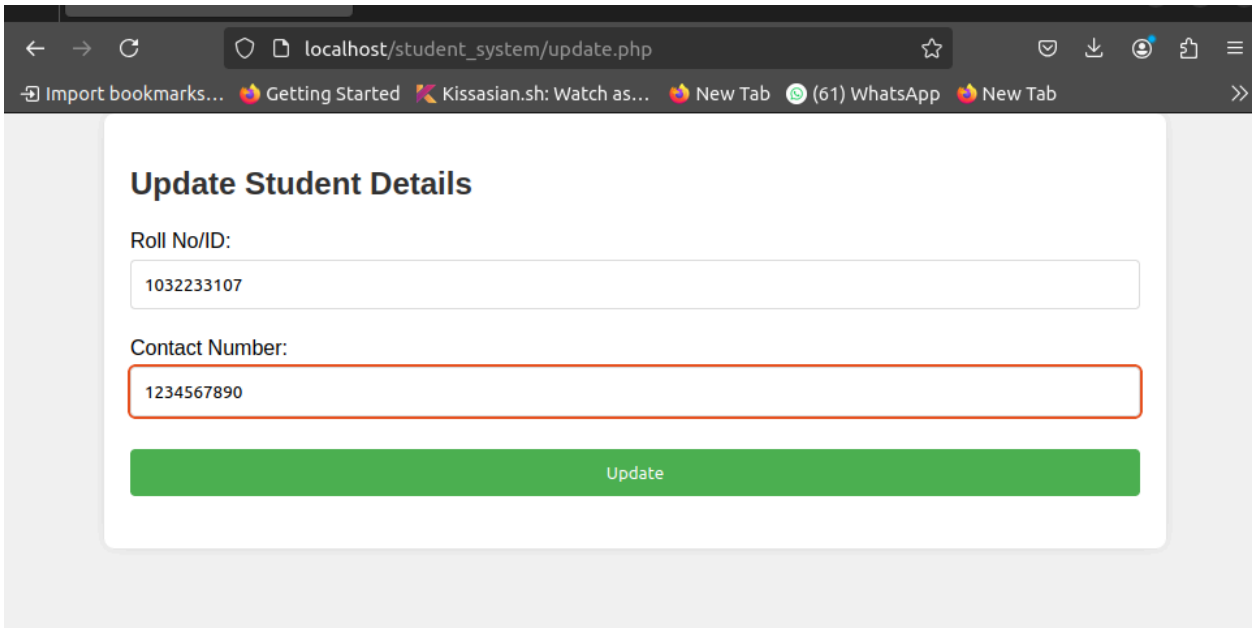


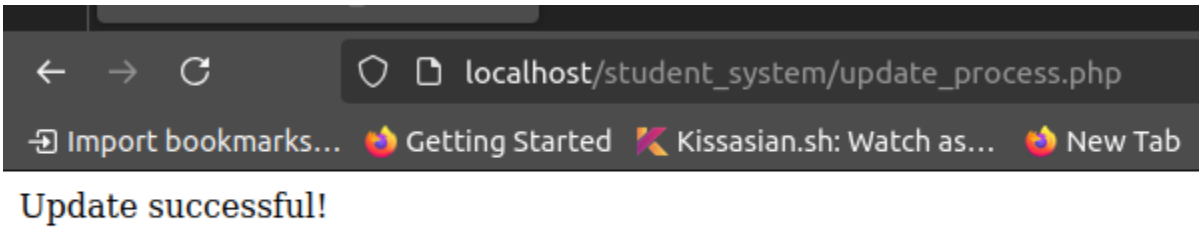


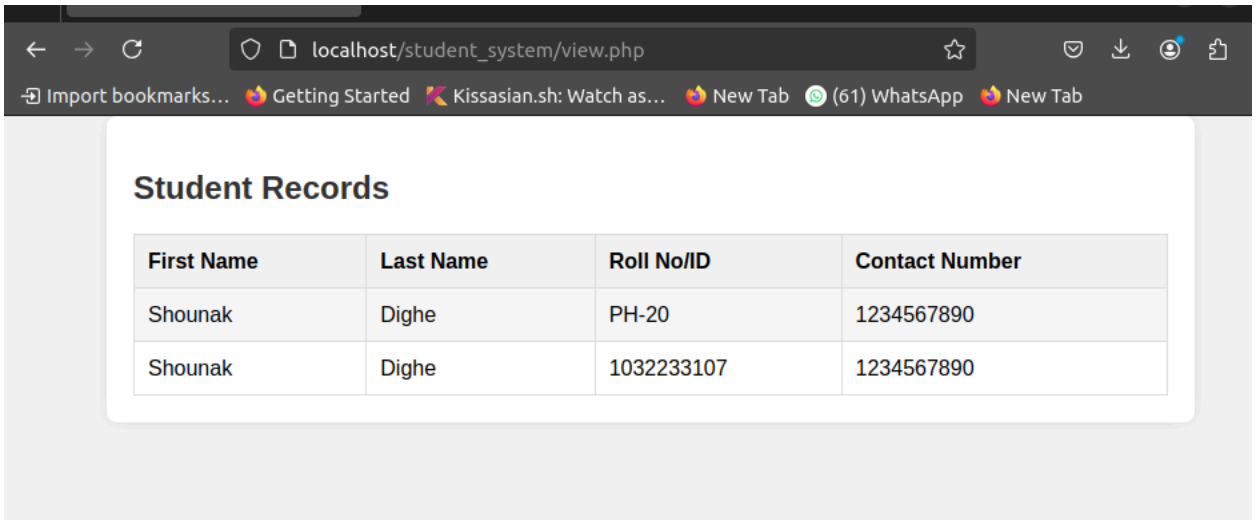
Registration successful!



Update Students Details

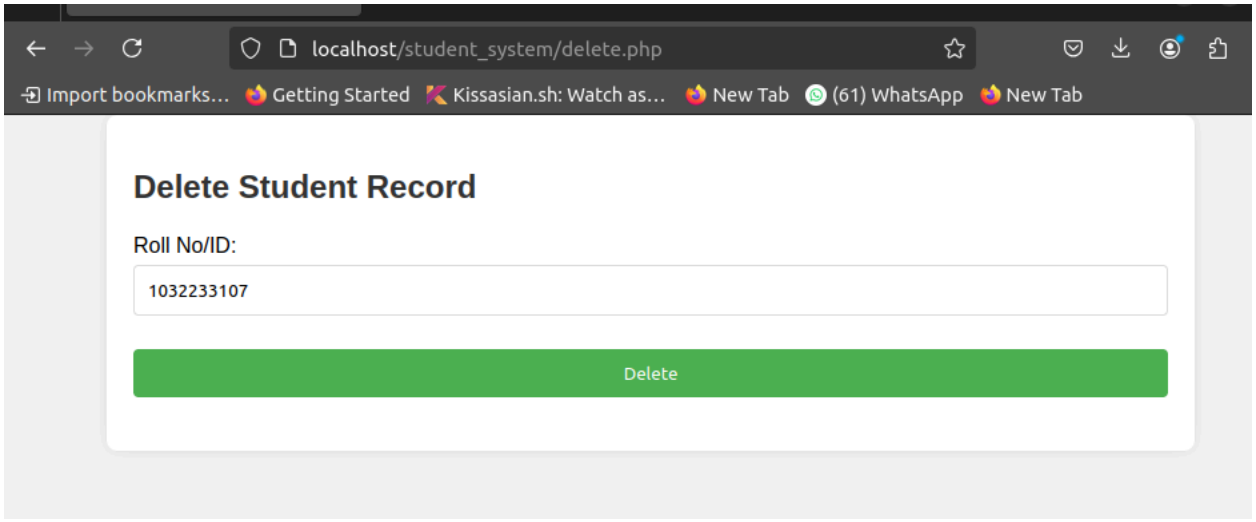


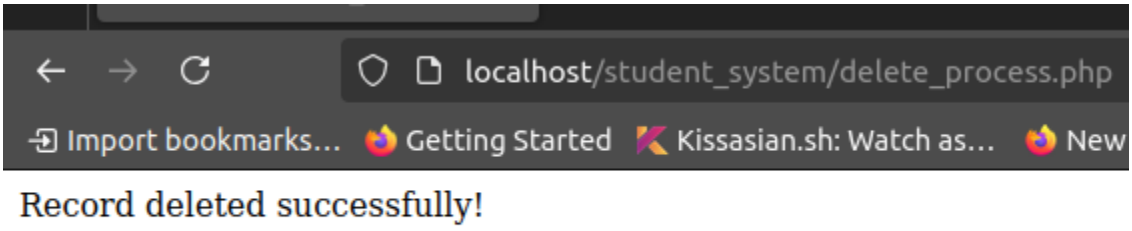


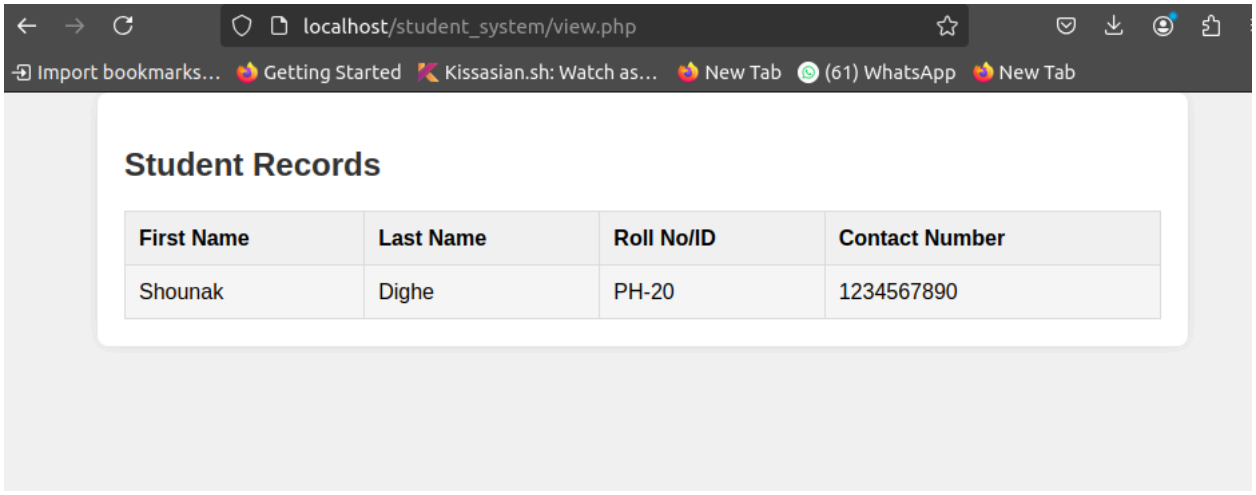


First Name	Last Name	Roll No/ID	Contact Number
Shounak	Dighe	PH-20	1234567890
Shounak	Dighe	1032233107	1234567890

Delete Student Record

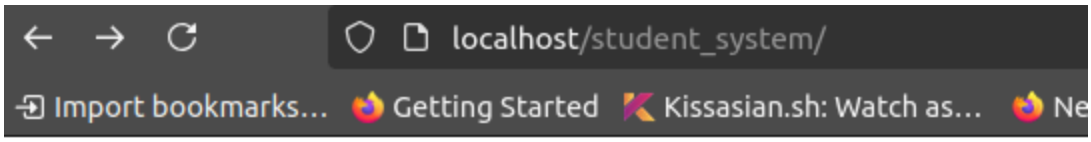






shounak@shounak: ~

```
shounak@shounak:~$ code  /var/www/html/student_system/
shounak@shounak:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 43
Server version: 8.0.39-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| student_db         |
| sys                |
+--------------------+
5 rows in set (0.02 sec)

mysql> USE student_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
+----------------------+
| Tables_in_student_db |
+----------------------+
| students             |
+----------------------+
1 row in set (0.01 sec)

mysql> DESCRIBE students;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| id             | int          | NO   | PRI | NULL    | auto_increment |
| first_name     | varchar(50)  | NO   |     | NULL    |                |
| last_name      | varchar(50)  | NO   |     | NULL    |                |
| roll_no        | varchar(20)  | NO   | UNI | NULL    |                |
| password       | varchar(255) | NO   |     | NULL    |                |
| contact_number | varchar(20)  | NO   |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
6 rows in set (0.01 sec)

mysql> SELECT * FROM students;
+----+------------+-----------+---------+-------------------------------------
----------------------+----------------+
| id | first_name | last_name | roll_no | password
                     | contact_number |
```



Index of /student_system

Name	Last modified	Size	Description
Parent Directory		-	
db.php	2024-09-14 16:47	247	
delete.php	2024-09-14 17:04	427	
delete_process.php	2024-09-14 16:48	390	
register.php	2024-09-14 17:03	1.7K	
register_process.php	2024-09-14 16:47	885	
styles.css	2024-09-14 17:03	1.1K	
update.php	2024-09-14 17:04	891	
update_process.php	2024-09-14 16:47	464	
view.php	2024-09-14 17:04	1.3K	

Apache/2.4.52 (Ubuntu) Server at localhost Port 80